APLIKASI SPEECH APPLICATION PROGRAMMING INTERFACE (SAPI) 5.1 SEBAGAI PERINTAH UNTUK PENGOPERASIAN APLIKASI BERBASIS WINDOWS

Abdusy Syarif[1], Tri Daryanto[2], M. Zaenal Arifin[3]
[1,2,3] *Prodi Teknik Informatika, Fakultas Ilmu Komputer,Universitas Mercu Buana*
Jl. Meruya Selatan, Kembangan, Jakarta Barat 11650
E-mail: abdusyarif@mercubuana.ac.id, trid_umb@yahoo.com, riefeen@yahoo.com

ABSTRAK
Salah satu penelitian pengenalan ucapan yang terkenal adalah yang dilakukan oleh Microsoft Corporation yang dikembangkan untuk sistem operasi Windows. Microsoft mengembangkan standar untuk mesin pengenalan ucapan, yaitu SAPI (Speech Application Programming Interface). SAPI memberikan kemampuan workstation untuk mengenali ucapan manusia sebagai masukan, dan membuat audio keluaran seperti suara manusia dari teks tertulis. Kemampuan ini menambahkan dimensi baru interaksi manusia dan komputer. Layanan pengenalan ucapan dapat digunakan untuk memperluas penggunaan komputer bagi mereka yang menemui bahwa mengetik terlalu sulit karena keterbatasan fisik. Penelitian ini bertujuan untuk mengembangkan Aplikasi Perintah Suara untuk mengoperasikan aplikasi berbasis Windows guna memudahkan pengguna yang mempunyai keterbatasan fisik. Model pengembangan perangkat lunak yang digunakan dalam penelitian ini adalah model waterfall. Penjelasan rinci tentang algoritma pengenalan ucapan berada di luar cakupan makalah ini.

Kata Kunci: Speech Application Programming Interface, SAPI, Pengenalan Suara

1. PENDAHULUAN

Cara interaksi antara manusia dan komputer saat ini, pada umumnya masih dilakukan secara tanpa lisan. Cara tersebut dilakukan menggunakan keyboard sebagai piranti utama untuk memasukkan data dan layar monitor sebagai piranti utama untuk menyajikan hasil komputasi. Dengan berkembangnya sistem operasi berbasis grafik telah menyebabkan digunakannya piranti mouse sebagai pelengkap keyboard.

Keinginan untuk membuat cara berinteraksi yang lebih alami menyebabkan perlunya dicari alternatif lain sebagai pengganti atau pelengkap piranti yang sudah ada. Manusia dalam berinteraksi dengan sesama manusia secara umum banyak dilakukan menggunakan ucapan. Dengan menerapkan cara interaksi antara sesama manusia tersebut diharapkan menjadi cara alternatif yang dapat dipakai untuk interaksi antara manusia dan komputer secara lisan layaknya interaksi antara manusia dengan manusia.

Dengan didukung oleh berkembangnya teknologi multimedia memungkinkan dilakukan penelitian dan pengembangan dalam mengolah komponen multimedia, sehingga bisa dimanfaatkan sebagai sarana interaksi antara manusia dan komputer. Komponen multimedia terus dilakukan penelitian dan yang sudah banyak dikembangkan dari segi pemanfaatannya adalah suara. Dengan memanfaatkan masukan yang berupa suara ini dimungkinkan untuk melakukan interaksi secara lisan melalui ucapan.

Cara interaksi menggunakan ucapan dianggap lebih mudah dan nyaman dilakukan. Untuk mewujudkan keinginan tersebut ada dua teknologi kunci yang diperlukan, yaitu sebagai berikut:

- Teknologi pengenalan ucapan (*speech recognition*). Teknologi ini diperlukan untuk mengenali setiap ucapan menjadi teks.
- Teknologi pensintesa ucapan (*speech synthesizer*). Teknologi ini diperlukan untuk mengucapkan informasi teks yang dihasilkan oleh komputer. Sistem seperti ini dikenal pula dengan istilah sistem *Text To Speech* (TTS).

Banyak penelitian yang telah dan sedang dilakukan untuk mendapatkan pengenalan ucapan yang cepat dan akurat. Salah satu yang terkenal adalah penelitian yang dilakukan oleh *Microsoft Corporation* yang dikembangkan untuk sistem operasi Windows. Selain mengembangkan mesin pengenalan ucapan, Microsoft juga mengembangkan standar untuk pengenalan ucapan yaitu *Speech Application Programming Interface* (SAPI).

Sehingga dengan adanya penelitian ini penulis membuat rumusam masalah bagaimana merancang sebuah aplikasi pengenalan suara yang dapat melakukan perintah pada sistem operasi windows

2. LANDASAN TEORI

2.1 Komponen Sistem Dialog Lisan

Bahasa alami (*natural language*) merupakan mekanisme komunikasi yang atraktif. Umumnya, komputer tidak dapat mengerti instruksi yang dituliskan dalam bahasa sehari-hari. Bahasa alami dapat mengerti masukan tertulis (*written input*) dan masukan ucapan (*speech input*) '(Dix, 2009)'. Namun masih ada kekurangan dalam hal kerancuan (*ambiguity*) pada aspek sintaksis dan semantik. Sistem dialog lisan terdiri dari beberapa komponen yang dibutuhkan supaya sistem dapat berfungsi dengan sukses, diantarnya pengenalan ucapan

(*speech recognition*) dan teks ke ucapan (*text to speech*) '(McTear, 2004)'. Teknologi pengenalan ucapan memungkinkan perangkat komputer yang dilengkapi dengan mikropon untuk menerjemahkan ucapan manusia '(Kumar, 2005)'.

2.2 Sistem Pengenalan Ucapan Pada SAPI

Setiap sistem pengenalan ucapan pada intinya memiliki suatu proses untuk mengenali ucapan manusia dan mengubahnya menjadi sesuatu yang dimengerti oleh komputer '(Amundsen,1996)'. Untuk itu sebuah komputer membutuhkan penerjemah. Penelitian algoritma pengenalan ucapan yang efektif dan model pengolahannya telah berlangsung hampir sejak komputer diciptakan. Setiap sistem pengenalan ucapan menggunakan empat operasi kunci untuk mendengarkan dan memahami ucapan manusia, yaitu:

a) Pemisahan kata (*word separation*), ini adalah proses menciptakan bagian terpisah (*discreet*) dari ucapan manusia. Setiap bagian dapat sebagai frasa yang besar atau kecil sebagai suku kata tunggal atau bagian kata.
b) Perbendaharaan kata (*vocabulary*), ini adalah daftar item suara yang dapat diidentifikasi oleh mesin pengenalan ucapan.
c) Pencocokan kata (*word matching*), ini adalah metode sistem pengenalan ucapan yang digunakan untuk mencari bagian suara di sistem perbendaharaan kata (bagian mesin pencari dari sistem).
d) Ketergantungan pembicara (*speaker dependence*), ini adalah sejauh mana mesin pengenalan ucapan bergantung pada nada vokal dan pola bicara individu.

Elemen terakhir dari mesin pengenalan ucapan adalah aturan tata bahasa (*grammar rule*). Aturan tata bahasa digunakan oleh perangkat lunak pengenalan ucapan untuk menganalisis masukan ucapan manusia dan di dalam proses berupaya untuk memahami apa yang seseorang katakan. Ada banyak tata bahasa, masing-masing terdiri dari satu set aturan pengucapan. Salah satunya adalah tata bahasa bebas konteks (*context-free grammars* atau disingkat CFG). Unsur-unsur utama CFG yaitu:

a. Kata-kata (*words*), daftar kata yang valid untuk diucapkan.
b. Aturan-aturan (*rules*), struktur ucapan untuk kata-kata digunakan.
c. Daftar (*lists*), daftar kata untuk digunakan dalam aturan.

CFG mampu menangani perbendaharaan kata yang berubah-ubah. Hal ini karena sebagian besar bangunan aturan dilakukan CFG untuk menyelesaikan daftar deklarasi dan kelompok kata yang sesuai dengan pola umum atau aturan umum. Setelah mesin pengenalan ucapan memahami aturan, sangat mudah untuk memperluas perbendaharaan kata dengan memperluas daftar kemungkinan anggota kelompok. Sebagai contoh, aturan dalam CFG:

<NameRule>=ALT("Mike","Curt","Sharon","Angelique")

<SendMailRule>=("Send Email to", <NameRule>)

Dalam contoh di atas, dua aturan telah ditetapkan. Aturan pertama, <NameRule>, menciptakan daftar nama yang mungkin. Aturan kedua, <SendMailRule>, menciptakan aturan yang bergantung pada <NameRule>. Dengan cara ini, CFG memungkinkan untuk membangun aturan tata bahasa sendiri sebagai pemrediksi bagaimana manusia akan berinteraksi dengan sistem. Bahkan lebih penting lagi, CFG memungkinkan untuk ekspansi yang mudah pada saat run-time. Karena cara operasi CFG berfokus pada daftar, memberikan kemudahan bagi pengguna untuk menambah daftar anggota. Pembangunan CFG yang berkualitas merupakan tantangan, namun sistem yang hanya perlu melakukan beberapa hal-hal (seperti memuat dan menjalankan program, eksekusi petunjuk sederhana, dan sebagainya) mudah disajikan menggunakan CFG. Namun, dalam rangka untuk melakukan tugas yang lebih kompleks atau tugas yang lebih luas, aturan tambahan diperlukan. Karena CFG dasar prediksinya pada aturan yang telah ditetapkan, CFG tidak baik untuk tugas-tugas seperti dikte, dimana perbendaharaan kata yang besar merupakan suatu hal yang paling penting.

2.3 Application Programming Interface (API) SAPI 5.1

Dua tipe dasar mesin SAPI adalah sistem teks ke ucapan (*Text-To-Speech*) dan pengenalan ucapan (*Speech Recognition*). Sistem TTS mensintesis string teks dan file ke dalam audio ucapan menggunakan suara sintetik. Pengenalan ucapan mengkonversi audio yang diucapkan manusia ke dalam string teks yang dapat dibaca dan file.

Aplikasi dapat mengontrol teks ke ucapan dengan menggunakan *Component Object Model* (COM) *ISpVoice*. Setelah aplikasi menciptakan suatu obyek *ISpVoice*, aplikasi hanya perlu untuk memanggil ISpVoice::Speak untuk menghasilkan keluaran ucapan dari beberapa data teks.

Sebuah aplikasi memiliki dua pilihan jenis mesin pengenalan ucapan (*ISpRecognizer*). Sebuah pengenalan bersama (*shared recognition*) yang mungkin bisa dibagi dengan aplikasi pengenalan ucapan lainnya dianjurkan untuk sebagian besar aplikasi pengenalan ucapan. Untuk membuat ISpRecoContext untuk ISpRecognizer bersama (*shared*), aplikasi hanya perlu melakukan panggilan COM CoCreateInstance pada komponen CLSID_SpSharedRecoContext. Untuk aplikasi server besar yang akan berjalan sendirian pada sebuah sistem, dan kinerja adalah kunci, mesin pengenalan ucapan InProc yang lebih sesuai. Dalam rangka menciptakan ISpRecoContext untuk ISpRecognizer

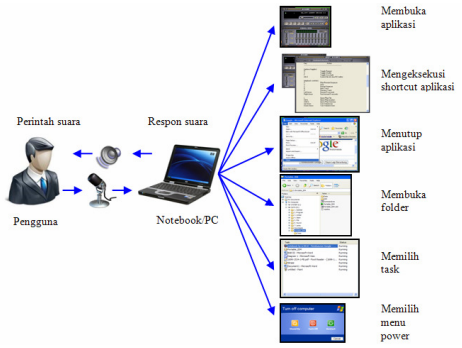
ISSN: 1907-5022

InProc, pertama aplikasi harus memanggil CoCreateInstance pada komponen CLSID_SpInprocRecoInstance untuk membuat sendiri ISpRecognizer InProc. Selanjutnya, aplikasi dapat memanggil ISpRecognizer::CreateRecoContext untuk mendapatkan sebuah ISpRecoContext.

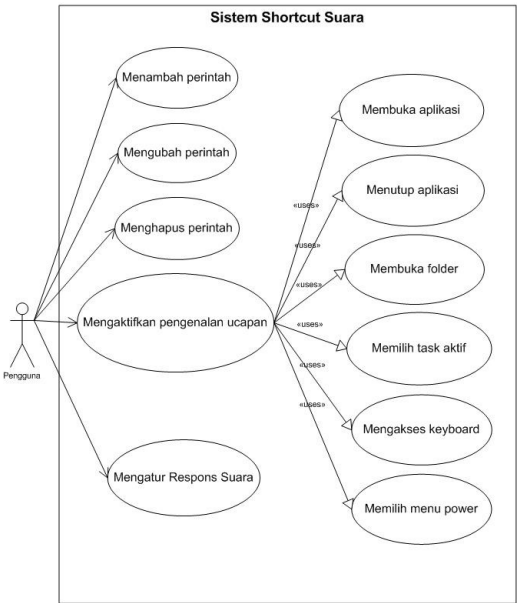
Akhirnya, sebuah aplikasi pengenalan ucapan harus membuat (*create*), memuat (*load*), dan mengaktifkan ISpRecoGrammar, yang pada dasarnya menunjukkan jenis ujaran untuk dikenali, yaitu tata bahasa dikte (*dictation*) atau perintah dan kontrol (*command and control*). Pertama, aplikasi ini menciptakan ISpRecoGrammar menggunakan ISpRecoContext::CreateGrammar. Kemudian, memuat aplikasi tata bahasa yang sesuai, baik dengan memanggil ISpRecoGrammar::LoadDictation untuk dikte atau salah satu dari cara ISpRecoGrammar:: LoadCmdxxx untuk perintah dan kontrol. Akhirnya, untuk mengaktifkan tata bahasa ini sehingga pengenalan ucapan dapat memulai aplikasi dengan memanggil ISpRecoGrammar::SetDictationState untuk dikte atau ISpRecoGrammar::SetRuleState atau ISpRecoGrammar::SetRuleIdState untuk perintah dan kontrol.

3. RANCANGAN SISTEM

Gambar 1 memperlihatkan rancangan sistem. Pada bagian masukan, seorang pengguna memberikan perintah suara (perintah ucapan). Suara tersebut diterima oleh perangkat mikropon untuk dirubah menjadi sinyal suara analog. Sinyal suara analog dari mikropon kemudian dimasukkan kedalam komputer (Notebook/PC) melalui bagian kartu suara (Sound Card). Sinyal suara analog yang diterima oleh kartu suara akan dirubah menjadi sinyal suara digital. Pada bagian proses, dengan menggunakan sistem SAPI 5.1 (*Speech Application Programming Interface*), sinyal suara digital yang dihasilkan dari kartu suara akan dipecah menjadi bagian suara terkecil (fonem). Fonem-fonem tersebut akan disusun sehingga menjadi data yang bisa dicocokkan dengan kata-kata dari string teks. Perintah yang bisa dicocokkan dengan string teks tersebut digunakan untuk melakukan pemanggilan fungsi eksekusi shortcut. Selain itu sistem SAPI 5.1 juga digunakan untuk mengubah string teks menjadi sinyal suara. Keluaran dari sistem berupa hasil fungsi eksekusi yang meliputi, membuka aplikasi, mengeksekusi shortcut aplikasi, menutup aplikasi, membuka folder, memilih task, serta memilih menu power. Keluaran dari sistem juga berupa suara yang merupakan respons dari sistem terhadap perintah-perintah yang diberikan kepada sistem. Keluaran suara ini disajikan kepada pengguna melalui perangkat speaker.



Gambar 1. Rancangan Sistem Aplikasi Perintah Suara Dengan Speech API



Gambar 2. Diagram Use Case Aplikasi

Masing-masing use case dalam Use Case Diagram pada Gambar 2 dapat dijelaskan sebagai berikut:

a) Menambah perintah, digunakan untuk menambahkan perintah yang baru ke dalam basis data sistem.
b) Mengubah perintah, digunakan untuk mengubah perintah yang telah disimpan dalam basis data sistem.
c) Menghapus perintah, digunakan untuk menghapus perintah tertentu dari basis data sistem.
d) Mengaktifkan pengenalan ucapan, digunakan untuk mengaktifkan mesin pengenalan ucapan agar sistem dapat menerima masukan suara.
e) Membuka aplikasi, digunakan untuk membuka aplikasi melalui perintah lisan.
f) Menutup aplikasi, digunakan untuk menutup aplikasi yang aktif melalui perintah lisan.
g) Membuka folder, digunakan untuk membuka folder dengan perintah lisan.

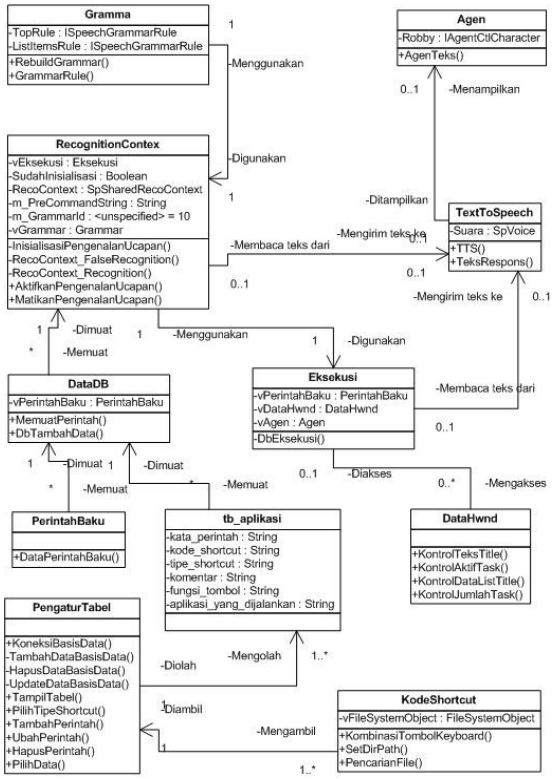
h) Memilih task aktif, digunakan untuk memilih task dari aplikasi yang aktif untuk ditempatkan pada posisi on top melalui perintah lisan.
i) Mengakses keyboard, digunakan untuk melakukan eksekusi dari shortcut yang berupa kombinasi tombol keybord dengan perintah lisan.
j) Memilih menu *power*, digunakan untuk memilih dan mengeksekusi pilihan menu *power* yang meliputi *shutdown*, *restart*, atau *logoff* melalui perintah lisan.

Masing-masing class dalam Class Diagram seperti terlihat pada Gambar 3, dapat dijabarkan dalam penjelasan dibawah ini.

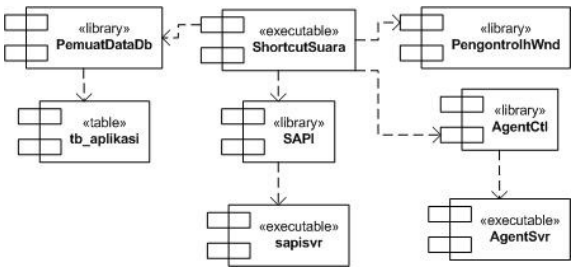
a. KodeShortcut, digunakan untuk membuat file aplikasi yang akan dioperasikan dan kombinasi shortcut tombol dari aplikasi yang akan diakses menjadi kode yang dapat dieksekusi oleh sistem. *Class* ini memiliki fungsi KombinasiTombolKeyboard(), SetDirPath() dan PencarianFile(). Fungsi KombinasiTombolKeyboard() untuk mengubah tombol keyboard ke dalam kode yang dapat digunakan oleh sistem untuk mengeksekusi tombol keyboard tanpa menekan fisik tombol. Fungsi SetDirPath() untuk menyimpan lokasi dari file aplikasi atau file folder yang diinginkan pengguna. Sedangkan fungsi PencarianFile() untuk melakukan pencarian file yang berjenis aplikasi (exe).
b. PengaturTabel, digunakan untuk mengatur data yang akan disimpan ke dalam tabel basis data. *Class* ini memiliki fungsi KoneksiBasisData(), TambahDataBasisData(), HapusDataBasisData(), UpdateDataBasisData(), TampilTabel(), PilihTipeShortcut(), TambahPerintah(), UbahPerintah(), HapusPerintah(), dan PilihData(). Fungsi KoneksiBasisData() untuk melakukan koneksi ke basis data. Fungsi TambahDataBasisData() untuk menambahkan perintah baru dan menyimpannya dalam tabel basis data. Fungsi HapusDataBasisData() untuk menghapus perintah dari basis data. Fungsi UpdateDataBasisData() untuk mengubah perintah dalam basis data. Fungsi TampilTabel() untuk menampilkan data dari basis data. Fungsi PilihTipeShortcut() untuk memilih jenis *shortcut* yang diinginkan oleh pengguna. Fungsi TambahPerintah() digunakan oleh pengguna untuk mengakses fungsi TambahDataBasisData(). Fungsi UbahPerintah() digunakan oleh pengguna untuk mengakses fungsi UpdateDataBasisData(). Fungsi HapusPerintah() digunakan pengguna untuk mengakses fungsi HapusDataBasisData(). Sedangkan fungsi PilihData() untuk memilih perintah yang akan dirubah atau dihapus.
c. tb_aplikasi, digunakan untuk menyimpan seluruh data perintah.
d. PerintahBaku, digunakan untuk memuat daftar perintah yang bersifat statis atau bersifat tetap. Fungsi dalam *class* ini yaitu DataPerintahBaku() untuk mengirimkan daftar perintah baku ke class DataDb.
e. DataDb, digunakan untuk memuat seluruh data perintah dari *class* PerintahBaku dan tb_aplikasi. Fungsi dalam *class* ini yaitu MemuatPerintah() dan DbTambahData(). Fungsi MemuatPerintah() untuk memuat perintah dari *class* PerintahBaku dan tb_aplikasi. Sedangkan fungsi DbTambahData() untuk menambahkan kata perintah baru pada saat sistem membuka aplikasi, menutup aplikasi atau membuka *folder*.
f. RecognitionContext, digunakan untuk melakukan pengenalan ucapan. Fungsi dalam *class* ini yaitu InisialisasiPengenalanUcapan(), RecoContext_FalseRecognition(), RecoContext_Recognition(), AktifkanPengenalanUcapan(), dan MatikanPengenalanUcapan(). Fungsi InisialisasiPengenalanUcapan() untuk melakukan konfigurasi sebelum pengenalan ucapan diaktifkan. Fungsi RecoContext_FalseRecognition() untuk menerima pemberitahuan bahwa pengenalan ucapan gagal melakukan pengenalan. Fungsi RecoContext_Recognition() untuk menerima hasil pengenalan perintah. Fungsi AktifkanPengenalanUcapan() untuk menjalankan sistem pengenalan ucapan. Fungsi MatikanPengenalanUcapan() untuk menghentikan mesin penegenalan ucapan.
g. Grammar, digunakan untuk mengaktifkan dan memuat tata bahasa dan aturan tata bahasa. Fungsi pada class ini yaitu RebuildGrammar() dan GrammarRule(). Fungsi RebuildGrammar() untuk memuat atau memuat ulang tata bahasa. Fungsi GrammarRule() untuk mengaktifkan aturan tata bahasa.
h. Eksekusi, digunakan untuk melakukan eksekusi kode shortcut. Fungsi dalam *class* ini yaitu DbEksekusi() untuk melakukan eksekusi kode shortcut yang dimuat dari basis data perintah.
i. DataHwnd, digunakan untuk mengamati setiap perubahan *handle window* (hWnd) dari aplikasi serta menyimpan hWnd dari setiap perubahan kejadian pada aplikasi. Fungsi dari *class* ini yaitu KontrolTaskTitle(), KontrolAktifTask(), KontrolDataListTitle(), dan KontrolJumlahTask(). Fungsi KontrolTeksTitle() untuk mengamati dan mencatat perubahan *task title* dari aplikasi. Fungsi KontrolAktifTask() untuk menempatkan task aplikasi pada posisi *on top* dengan memanggil *task title*. Fungsi KontrolDataListTitle() untuk menyimpan seluruh daftar *title* aplikasi. Fungsi KontrolJumlahTask() untuk mengatur penambahan daftar perintah baru berdasarkan jumlah *task* dari aplikasi yang aktif.
j. TextToSpeech, digunakan untuk merubah string teks ke suara. Fungsi dari *class* ini yaitu TTS() dan TeksRespons(). Fungsi TTS() untuk merubah string teks ke ucapan. Fungsi TeksRespons()

untuk memilih respon dari sistem, disampaikan melalui suara atau pesan teks.

k. Agen, digunakan untuk menampilkan animasi aplikasi MS Agen. Fungsi dari *class* ini yaitu AgenTeks() untuk menampilkan respon teks dari sistem.



Gambar 3. Class Diagram



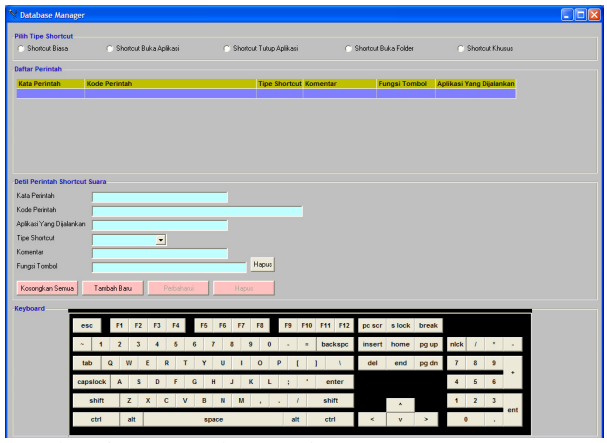
Gambar 4. *Component Diagram*

Rincian dari masing-masing komponen pada component diagram pada Gambar 4 tersebut dapat dijelaskan sebagai berikut:

a. tb_aplikasi, merupakan komponen berjenis tabel. Basis data dari sistem ini hanya terdiri dari sebuah tabel saja yaitu tb_aplikasi tersebut. Komponen ini terhubung dengan *class* PengaturTabel dan KodeShortcut.
b. PemuatDataDb, merupakan komponen berjenis *library*. Komponen ini berfungsi untuk menampung seluruh daftar perintah dari basis data sebelum digunakan. Komponen ini terhubung dengan class DataDb dan PerintahBaku.
c. ShortcutSuara, merupakan komponen berjenis *executable*. Komponen ini merupakan komponen utama yang berjalan pada saat run time untuk melakukan pengenalan ucapan dan eksekusi. Komponen ini terbentuk dari *class Grammar, RecognitionContext* dan *class* Eksekusi.
d. SAPI, merupakan komponen berjenis *library*. File *library* tersebut bernama sapi.dll. Komponen ini digunakan oleh *class Grammar, RecognitionContext* dan *TextToSpeech.*
e. sapisvr, merupakan komponen *executable* yang aktif secara otomatis pada saat sapi.dll dijalankan. Komponen ini aktif karena jenis mesin pengenalan yang digunakan menggunakan konteks pengenalan yang bisa digunakan untuk berbagi pakai (*Shared Recognition Context*).
f. PengontrolHwnd, merupakan komponen berjenis *library*. Komponen ini untuk mengamati hWnd dan menyimpannya. Komponen ini terhubung dengan *class DataHwnd.*
g. AgenCtl, merupakan komponen berjenis *library*. Komponen ini untuk mengaktifkan aplikasi MS Agen. Komponen ini digunakan oleh *class Agen*.
h. AgenSvr, merupakan komponen berjenis *executable*. Komponen ini berjalan pada saat *run time* ketika MS Agen aktif.

4. IMPLEMENTASI DAN ANALISA PENGUJIAN

4.1 Implementasi dan Pengujian



Gambar 5. Antarmuka *Database Manager*

Antarmuka *Database Manager* terdiri dari empat bingkai (frame). Bingkai yang pertama adalah yang paling atas. Bingkai pertama berisi pilihan untuk menu tipe *shortcut*, bingkai kedua berisi komponen untuk menampilkan data perintah dari basis data sesuai jenis *shortcut* yang dipilih, bingkai ketiga berisi borang untuk memasukkan data perintah, dan bingkai keempat berisi komponen yang membentuk papan ketik untuk memasukkan *shortcut* tombol.

Pengujian mesin pengenalan ucapan dari sistem Perintah Suara dilakukan oleh dua orang penguji, seorang pria dan seorang wanita dewasa. Pengujian dilakukan dengan melakukan pengucapan perintah

 ISSN: 1907-5022

sebanyak sepuluh perintah yang berbeda, masing-masing diulangi sebanyak dua puluh kali secara acak dan kemudian dihitung jumlah perintah yang dikenali secara benar maupun salah. Selain itu dihitung pula jumlah ucapan perintah yang tidak dikenali. Pengujian dilakukan dalam dua tahap, sebelum penguji melakukan pelatihan dan sesudah penguji melakukan pelatihan. Kedua pengujian tersebut dilakukan untuk menentukan persentase keakuratan pengenalan ucapan.

Tabel 1. Hasil Pengujian Mesin Pengenalan Ucapan Sebelum Pelatihan

Perintah	Hasil pengenalan perintah							
	Pria		Wanita		Jumlah		Prosentase	
	B	G	B	G	B	G	B	G
Open winamp	19	1	15	5	34	6	85,00	15,00
Play winamp	20	0	19	1	39	1	97,50	2,50
Stop winamp	16	4	17	3	33	7	82,50	17,50
Close winamp	20	0	16	4	36	4	90,00	10,00
Open notepad	17	3	20	0	37	3	92,50	7,50
Open file	14	6	19	1	33	7	82,50	17,50
Cancel	19	1	18	2	37	3	92,50	7,50
Close notepad	15	5	18	2	33	7	82,50	17,50
Open database manager	15	5	15	5	30	10	75,00	25,00
Close database manager	18	2	17	3	35	5	87,50	12,50
Nilai rata-rata							86,75	13,25

Tabel 1 berisi hasil pengujian yang dilakukan sebelum penguji melakukan pelatihan. Hasil gagal (G) merupakan jumlah dari hasil salah dan hasil tidak dikenali.

Tabel 2. Hasil Pengujian Mesin Pengenalan Ucapan Setelah Pelatihan

Perintah	Hasil pengenalan perintah							
	Pria		Wanita		Jumlah		Prosentase	
	B	G	B	G	B	G	B	G
Open winamp	19	1	16	4	35	5	87,50	12,50
Play winamp	20	0	19	1	39	1	97,50	2,50
Stop winamp	19	1	18	2	37	3	92,50	7,50
Close winamp	19	1	18	2	37	3	92,50	7,50
Open notepad	19	1	20	0	39	1	97,50	2,50
Open file	14	6	20	0	34	6	85,00	15,00
Cancel	19	1	20	0	39	1	97,50	2,50
Close notepad	20	0	19	1	39	1	97,50	2,50
Open database manager	19	1	19	1	38	2	95,00	5,00
Close database manager	19	1	19	1	38	2	95,00	5,00
Nilai rata-rata							93,75	6,25

4.2 Analisa Pengujian

Pada pengujian yang dilakukan terhadap Aplikasi Perintah Suara, yaitu pengujian yang dilakukan terhadap mesin pengenalan ucapan pada Tabel 1 dan Tabel 2, maka dapat dianalisa bahwa Aplikasi Perintah Suara dapat digunakan tanpa melakukan pelatihan terlebih dahulu dan akan lebih baik bila digunakan setelah melakukan pelatihan. Tingkat keakuratan pengenalan ucapan pada pengujian sebelum pelatihan adalah 86,75% sedangkan tingkat keakuratan pada pengujian setelah pelatihan adalah 93,75%, ini berarti bahwa pengaruh dari luar, yaitu pelatihan yang dilakukan pengguna akan mempengaruhi kinerja sistem dalam melakukan pengenalan ucapan sehingga meningkatkan keakuratan pengenalan ucapan sebesar 7,00%.

Peningkatan keakuratan mesin pengenalan ucapan pada pengujian setelah pelatihan terjadi karena pelatihan yang dilakukan pengguna akan menurunkan tingkat kegagalan. Aplikasi Perintah Suara ini menggunakan mode perintah dan kontrol (*command and control*) yang berarti bahwa kata yang dikenali hanya kata yang telah terdefinisi dalam basis data yang jumlahnya sedikit, maka tingkat kegagalan akan cukup tinggi yaitu 13,25% pada pengujian tanpa pelatihan dan 6.25% pada pengujian setelah pelatihan.

5. KESIMPULAN

Kesimpulan yang kami ambil adalah :

a) Sistem *Speech API* mudah untuk diimplementasikan dalam pembuatan program yang memanfaatkan sistem pengenalan ucapan maupun sistem yang mengkonversi dari teks ke ucapan karena mengurangi kerumitan dalam hal kode program.
b) Aplikasi perintah suara ini dapat digunakan untuk mengoperasikan aplikasi lain karena aplikasi ini berada di luar aplikasi yang dioperasikan.
c) Aplikasi perintah suara ini dapat membantu pengguna komputer yang mengalami kesulitan dalam mengoperasikan komputer menggunakan *keyboard* dan *mouse*.
d) Kemampuan mengenali perintah dari aplikasi perintah suara ini terpengaruh oleh kondisi sekitar yang bising.
e) Aplikasi perintah suara ini dapat digunakan tanpa melakukan pelatihan terlebih dahulu.
f) Pelatihan yang dilakukan pengguna sebelum menggunakan aplikasi perintah suara ini akan meningkatkan keakuratan dalam mengenali perintah.
g) Hasil pengujian menunjukkan bahwa aplikasi ini lebih optimal penggunaannya jika melalui tahap pelatihan, yaitu memiliki tingkat keberhasilan mencapai 93,75%.
h) Aplikasi perintah suara ini hanya mampu untuk mengoperasikan aplikasi berdasarkan *shortcut* tombol dari aplikasi tersebut, sedangkan untuk pengetikan dan pembacaan file dokumen tidak mampu ditangani oleh aplikasi ini.

Saran yang dapat diajukan untuk pengembangan dan perbaikan sistem ini adalah :

- Penggunaan mikropon dengan kualitas yang bagus dapat mengurangi tingkat kesalahan dari sistem.
- Untuk menjaga kestabilan mesin pengenalan dari pengaruh kondisi sekitar, sistem harus dikembangkan untuk mampu mengatasi kebisingan sekitar yang ikut masuk ke dalam sistem melalui mikropon.
- Sistem ini akan sempurna bila dilengkapi dengan kemampuan pengetikan dan pembacaan file teks.

PUSTAKA

Amundsen, Michael C. MAPI, (1996). *SAPI, and TAPI Developer's Guide*, Indianapolis:Sams Publishing.

Dix, Alan , Janet Finlay, Gregory D. Abowd, and Russel Beale. (2009). *Human-Computer Interaction,* Third Edition, New Delhi:Pearson Education

Kumar, Rajendra. (2005). *Human Computer Interaction*, New Delhi:Firewall Media.

Hadi, Rahadian. (2005). *Penduan Bagi Pemrogram Microsoft Windows Common Controls*, Jakarta:Elex Media Komputindo.

Hendrayudi. (2009). *VB 2008 untuk Berbagai Keperluan Programming*, Jakarta:Elex Media Komputindo.

McTear, Michael F. (2004). *Spoken Dialogue Technology Toward the Conversatioanl User Interface*, London:Springer.

Yung, Kok. 128. (2006). *Teknik Profesional Windows Vista*, Jakarta:Elex Media Komputindo.